Python Training Course

Sami-Matias Niemi
smn2@mssl.ucl.ac.uk
Euclid VIS Instrument Scientist

MSSL - UCL, 1st of October 2012

Time	Topic
09:30 - 10:30	Python basics
15 mins	Coffee break
10:45 - 12:00	Numerical arrays
12:00 - 12:30	Astronomy specific packages
60 mins	Lunch break
13:30 - 14:15	Scientific analysis
14:15 - 15:00	Data access and processing
15 mins	Coffee break
15:15 - 16:00	Image processing
16:00 - 16:45	Overflow / Demo session
16:45 - 17:30	Speeding things up [if time]

Can I play as well?

- Core Python is pre-installed in Linux (Debian, Ubuntu, etc.) and OSX.
- Many ways to install 3rd party packages (easy_install, pip, homebrew, etc.).
  - One of the easiest way is, however, to use Enthought build:
    - http://enthought.com/products/epd.php
    - Free for Academics!
- You can also log in to msslae, a8, etc.

What is Python used for?

- Searching: Google search engine (+ C&Java).
- Maps: Yahoo maps
- Graphics: Industrial Light and Magic, Walt Disney Feature Animation...
- Science: National Weather Service, NASA, STScI (HST&JWST)...
- Software development: e.g. Red Hat installer
- Games: Battlefield 2, Civilization 4...

I have used Python for

- Euclid-VIS image simulator:
  - from HST images to mock observations.
- Spectral and imaging data reduction pipeline:
  - ACS polarimetry and WFC3IR slitless grism pipelines;
  - Ground based image slicer spectroscopic data reduction pipeline.
- All analysis and figures in my papers, proposals, etc. since 2008.

So others use Python,
why should I care?

Why Python?

- Free & Portable (Linux, OSX, Windows...)
- Easy to learn yet powerful
- Reduced development time:
  - No compilation phase
  - Integration with C, Fortran, and Java is easy.
  - Widely used in Astronomical community:
    - HST, Herschel, Chandra, ALMA, JWST, etc.
  - Many 3rd party packages...

Python Package Index

- http://pypi.python.org/pypi
- Software repository listing ~~24206~~ 24343 packages (as of ~~25th~~ 30th of Sep 2012).

Index of Packages Matching 'astronomy'

Not Logged In

Login
Register
Lost Login?
Use OpenID

Package	Weight*	Description
astropy 0.1	3	Community-developed python astronomy tools
ephem 3.7.5.1	3	Scientific-grade astronomy routines
novas 3.1	3	The United States Naval Observatory NOVAS astronomy library
pyephem 3.7.5.1	3	Scientific-grade astronomy routines
BOTEC 0.3	2	A simple astrophysics and orbital mechanics simulator
CosmoloPy 0.1.104	2	a cosmology package for Python.
mirpy 0.2.0	2	Pure-python miriad CLI wrapper.
poppy 0.2.8	2	Physical optics propagation (wavefront diffraction) for optical simulations, particularly of telescopes.
pywcs 1.10-4.7	2	Python wrappers to WCSLIB
pywcsgrid2 0.1b2	2	pywcsgrid2 is a python module to be used with matplotlib for displaying astronomical fits images
yt 2.4	2	An analysis and visualization toolkit for Astrophysical simulations, focusing on Adaptive Mesh Refinement data from Enzo, Orion, FLASH, and others.

Refinement data from Enzo, Orion, FLASH, and others.

...a few worth mentioning

- NumPy, SciPy, matplotlib, ipython, PyFITS, astropy, ATpy, Kapteyn, PyWCS, SymPy, scikits, RPy2, PyMC, emcee, PyIDL, APLpy, astLib, statsmodels, MySQLdb, PIL, PyCUDA, numexpr, yt, mplstereonet, h5py, Patsy, SfePy, SQLAlchemy, TKinter, PyGtk, PyQt, Mutagen, Django, numexpr, PyPy...
  - but too many to fit on one slide.
- Examples will use some of the above.

Powerful Inbuilt Functions

- Date and time (datetime, calendar)
- File and directory access (glob, shutil)
- OS service (os, sys, io, time)
- OS service ext. (thread, mmap)
- Math and complex numbers (math, cmath, fractions, decimals)
- Data persistence (cPickle, whichdb, sqlite3)
- Data compression (zlib, gzip, tarfile)

Python Syntax

- Designed to be highly readable.
- Indentation is crucial!
  - No keywords like "begin", "endif" or {}
- No need to declare variable types.
  - public static final double c = 299792458.0D; [JAVA]
- Some reserved words:
  - and, as, assert, break, class, for, if, import..

Python Basics

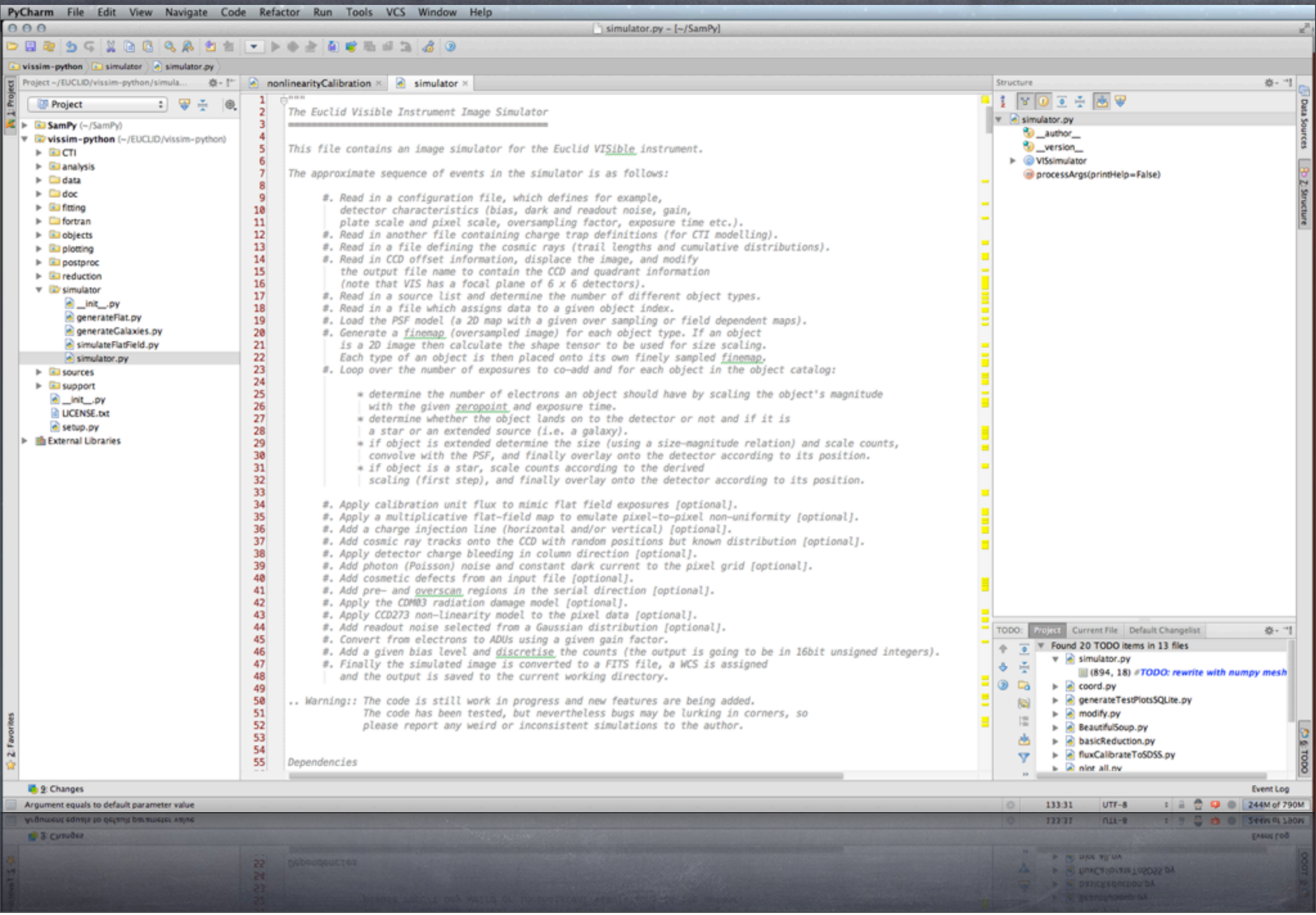
- You can write "scripts":
  - Files containing complete "programs" ready for execution.
- or use an interactive session:
  - Python interpreter
  - iPython (shell + Python):
    - provides shell commands, tab completion, history, etc.

Demo (i)python from a shell and then open the first notebook (Basics)...

Some Tools

- Development Environment:
  - E.g. PyCharm
- Documenting your code:
  - E.g. Sphinx
    - Directly from code to html, PDFLaTex
- Version control:
  - GIT, Mercurial, SVN, etc.

PyCharm - http://www.jetbrains.com/pycharm/



The file provides a function that returns VIS related information such as pixel size, dark current, gain, zeropoint, and sky background.

requires NumPy

requires numexpr

author Sami-Matias Niemi

contact smn2@mssl.ucl.ac.uk

version 0.4

`support.VISinstrumentModel.`**`CCDnonLinearityModel`**(*data*)

This function provides a non-linearity model for a VIS CCD273.

The non-linearity is modelled based on the results presented in MSSL/Euclid/TR/12001 issue 2. Especially Fig. 5.6, 5.7, 5.9 and 5.10 were used as an input data. The shape of the non-linearity is assumed to follow a parabola (although this parabola has a break, see the note below). The MSSL report indicates that the residual non-linearity is on the level of +/-25 DN or about +/- 0.04 per cent over the measured range. This function tries to duplicate this effect.

Note: There is a break in the model around 22000e. This is because the non-linearity measurements performed thus far are not extremely reliable below 10ke (< 0.5s exposure). However, the assumption is that at low counts the number of excess electrons appearing due to non-linearity should not be more than a few.

Parameters data (*float, int or ndarray*) – data to which the non-linearity model is being applied to

---

Returns input data after conversion with the non-linearity model

Return type float or ndarray

Exposure Time Calculator

Calculating Exposure Times and Limiting Magnitude

This file provides a simple functions to calculate exposure times or limiting magnitudes.

requires: NumPy
requires: SciPy
requires: matplotlib
version: 0.3
author: Sami-Matias Niemi
contact: smn2@mssl.ucl.ac.uk

---

analysis.ETC.**SNR**(*info, magnitude=24.5, exptime=565.0, exposures=3, galaxy=True, background=True, diginoise=True*)

Calculates the signal-to-noise ratio for an object of a given magnitude in a given exposure time and a number of exposures.

Parameters:
- **info** (*dict*) – instrumental information such as zeropoint and background
- **magnitude** (*float or ndarray*) – input magnitude of an object(s)
- **exptime** (*float*) – exposure time [seconds]
- **exposures** (*int*) – number of exposures [default = 3]
- **galaxy** (*boolean*) – whether the exposure time should be calculated for an average galaxy or a star. If galaxy=True then the fraction of flux within an aperture is lower than in case of a point source.
- **background** (*boolean*) – whether to include background from sky, instrument, and dark current [default=True]
- **diginoise** (*boolean*) – if the readout noise is undersampled or poorly resolved then the effective readout noise should be used [default = True]

Returns: signal-to-noise ratio

Return type: float or ndarray

---

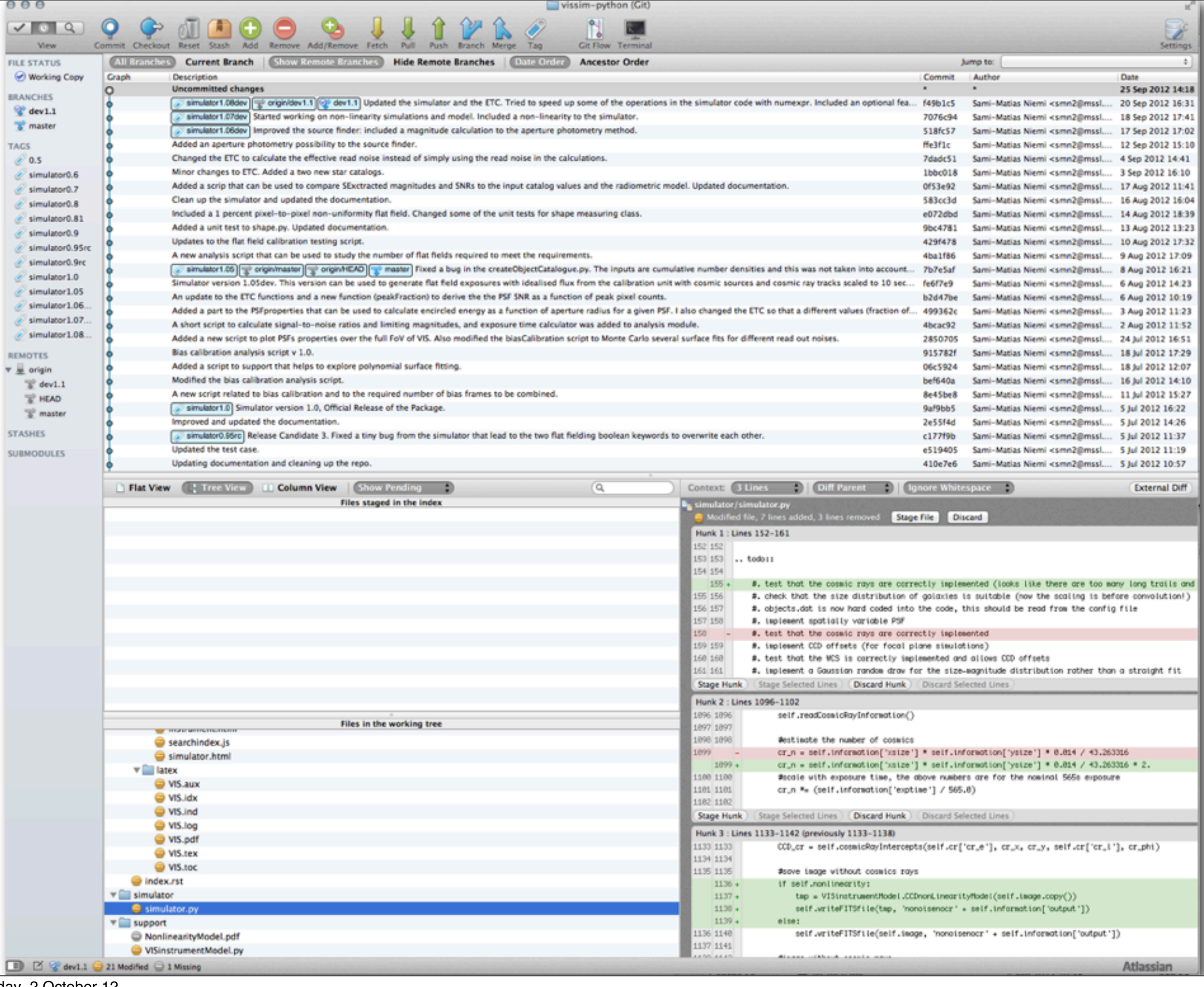
analysis.ETC.**SNRproptoPeak**(*info, exptime=565.0, exposures=1, diginoise=True*)

Calculates the relation between the signal-to-noise ratio and the electrons in the peak pixel.

Parameters:
- **info** (*dict*) – instrumental information such as zeropoint and background
- **exptime** (*float*) – exposure time [seconds]
- **exposures** (*int*) – number of exposures [default = 1]
- **diginoise** (*boolean*) – if the readout noise is undersampled or poorly resolved then the effective readout noise should be used [default = True]

Returns: signal-to-noise ratio

Return type: float or ndarray


vissim-python (Git)
View
Commit
Checkout
Reset
Stash
Add
Remove
Add/Remove
Fetch
Pull
Push
Branch
Merge
Tag
Git Flow
Terminal
Settings
FILE STATUS
Working Copy
BRANCHES
dev1.1
master
TAGS
0.5
simulator0.6
simulator0.7
simulator0.8
simulator0.81
simulator0.9
simulator0.95rc
simulator0.9rc
simulator1.0
simulator1.05
simulator1.06...
simulator1.07...
simulator1.08...
REMOTES
origin
dev1.1
HEAD
master
STASHES
SUBMODULES
All Branches
Current Branch
Show Remote Branches
Hide Remote Branches
Date Order
Ancestor Order
Jump to:
Graph
Description
Commit
Author
Date
Uncommitted changes
25 Sep 2012 14:18
simulator1.08dev
origin/dev1.1
dev1.1
Updated the simulator and the ETC. Tried to speed up some of the operations in the simulator code with numexpr. Included an optional fea...
f49b1c5
Sami-Matias Niemi <smn2@mssl....
20 Sep 2012 16:31
simulator1.07dev
Started working on non-linearity simulations and model. Included a non-linearity to the simulator.
7076c94
Sami-Matias Niemi <smn2@mssl....
18 Sep 2012 17:41
simulator1.06dev
Improved the source finder: included a magnitude calculation to the aperture photometry method.
518fc57
Sami-Matias Niemi <smn2@mssl....
17 Sep 2012 17:02
Added an aperture photometry possibility to the source finder.
ffe3f1c
Sami-Matias Niemi <smn2@mssl....
12 Sep 2012 15:10
Changed the ETC to calculate the effective read noise instead of simply using the read noise in the calculations.
7dadc51
Sami-Matias Niemi <smn2@mssl....
4 Sep 2012 14:41
Minor changes to ETC. Added a two new star catalogs.
1bbc018
Sami-Matias Niemi <smn2@mssl....
3 Sep 2012 16:10
Added a scrip that can be used to compare SExctracted magnitudes and SNRs to the input catalog values and the radiometric model. Updated documentation.
0f53e92
Sami-Matias Niemi <smn2@mssl....
17 Aug 2012 11:41
Clean up the simulator and updated the documentation.
583cc3d
Sami-Matias Niemi <smn2@mssl....
16 Aug 2012 16:04
Included a 1 percent pixel-to-pixel non-uniformity flat field. Changed some of the unit tests for shape measuring class.
e072dbd
Sami-Matias Niemi <smn2@mssl....
14 Aug 2012 18:39
Added a unit test to shape.py. Updated documentation.
9bc4781
Sami-Matias Niemi <smn2@mssl....
13 Aug 2012 13:23
Updates to the flat field calibration testing script.
429f478
Sami-Matias Niemi <smn2@mssl....
10 Aug 2012 17:32
A new analysis script that can be used to study the number of flat fields required to meet the requirements.
4ba1f86
Sami-Matias Niemi <smn2@mssl....
9 Aug 2012 17:09
simulator1.05
origin/master
origin/HEAD
master
Fixed a bug in the createObjectCatalogue.py. The inputs are cumulative number densities and this was not taken into account....
7b7e5af
Sami-Matias Niemi <smn2@mssl....
8 Aug 2012 16:21
Simulator version 1.05dev. This version can be used to generate flat field exposures with idealised flux from the calibration unit with cosmic sources and cosmic ray tracks scaled to 10 sec...
fe6f7e9
Sami-Matias Niemi <smn2@mssl....
6 Aug 2012 14:23
An update to the ETC functions and a new function (peakFraction) to derive the the PSF SNR as a function of peak pixel counts.
b2d47be
Sami-Matias Niemi <smn2@mssl....
6 Aug 2012 10:19
Added a part to the PSFproperties that can be used to calculate encircled energy as a function of aperture radius for a given PSF. I also changed the ETC so that a different values (fraction of...
499362c
Sami-Matias Niemi <smn2@mssl....
3 Aug 2012 11:23
A short script to calculate signal-to-noise ratios and limiting magnitudes, and exposure time calculator was added to analysis module.
4bcac92
Sami-Matias Niemi <smn2@mssl....
2 Aug 2012 11:52
Added a new script to plot PSFs properties over the full FoV of VIS. Also modified the biasCalibration script to Monte Carlo several surface fits for different read out noises.
2850705
Sami-Matias Niemi <smn2@mssl....
24 Jul 2012 16:51
Bias calibration analysis script v 1.0.
915782f
Sami-Matias Niemi <smn2@mssl....
18 Jul 2012 17:29
Added a script to support that helps to explore polynomial surface fitting.
06c5924
Sami-Matias Niemi <smn2@mssl....
18 Jul 2012 12:07
Modified the bias calibration analysis script.
bef640a
Sami-Matias Niemi <smn2@mssl....
16 Jul 2012 14:10
A new script related to bias calibration and to the required number of bias frames to be combined.
8e45be8
Sami-Matias Niemi <smn2@mssl....
11 Jul 2012 15:27
simulator1.0
Simulator version 1.0, Official Release of the Package.
9af9bb5
Sami-Matias Niemi <smn2@mssl....
5 Jul 2012 16:22
Improved and updated the documentation.
2e55f4d
Sami-Matias Niemi <smn2@mssl....
5 Jul 2012 14:26
simulator0.95rc
Release Candidate 3. Fixed a tiny bug from the simulator that lead to the two flat fielding boolean keywords to overwrite each other.
c177f9b
Sami-Matias Niemi <smn2@mssl....
5 Jul 2012 11:37
Updated the test case.
e519405
Sami-Matias Niemi <smn2@mssl....
5 Jul 2012 11:19
Updating documentation and cleaning up the repo.
410e7e6
Sami-Matias Niemi <smn2@mssl....
5 Jul 2012 10:57
Flat View
Tree View
Column View
Show Pending
Files staged in the index
Files in the working tree
searchindex.js
simulator.html
latex
VIS.aux
VIS.idx
VIS.ind
VIS.log
VIS.pdf
VIS.tex
VIS.toc
index.rst
simulator
simulator.py
support
NonlinearityModel.pdf
VISinstrumentModel.py
Context:
3 Lines
Diff Parent
Ignore Whitespace
External Diff
simulator/simulator.py
Modified file, 7 lines added, 3 lines removed
Stage File
Discard
Hunk 1 : Lines 152-161
.. todo::
#. test that the cosmic rays are correctly implemented (looks like there are too many long trails and
#. check that the size distribution of galaxies is suitable (now the scaling is before convolution!)
#. objects.dat is now hard coded into the code, this should be read from the config file
#. implement spatially variable PSF
#. test that the cosmic rays are correctly implemented
#. implement CCD offsets (for focal plane simulations)
#. test that the WCS is correctly implemented and allows CCD offsets
#. implement a Gaussian random draw for the size-magnitude distribution rather than a straight fit
Stage Hunk
Stage Selected Lines
Discard Hunk
Discard Selected Lines
Hunk 2 : Lines 1096-1102
self.readCosmicRayInformation()
#estimate the number of cosmics
cr_n = self.information['xsize'] * self.information['ysize'] * 0.014 / 43.263316
cr_n = self.information['xsize'] * self.information['ysize'] * 0.014 / 43.263316 * 2.
#scale with exposure time, the above numbers are for the nominal 565s exposure
cr_n *= (self.information['exptime'] / 565.0)
Stage Hunk
Stage Selected Lines
Discard Hunk
Discard Selected Lines
Hunk 3 : Lines 1133-1142 (previously 1133-1138)
CCD_cr = self.cosmicRayIntercepts(self.cr['cr_e'], cr_x, cr_y, self.cr['cr_l'], cr_phi)
#save image without cosmics rays
if self.nonlinearity:
tap = VISinstrumentModel.CCDnonLinearityModel(self.image.copy())
self.writeFITSfile(tap, 'nonoisenocr' + self.information['output'])
else:
self.writeFITSfile(self.image, 'nonoisenocr' + self.information['output'])
dev1.1
21 Modified
1 Missing
Atlassian

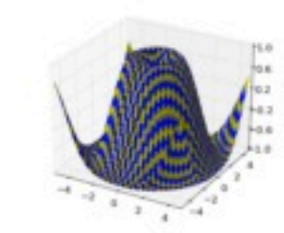
Numerical Arrays

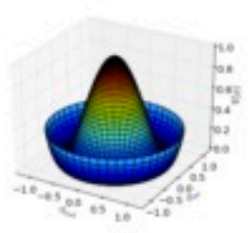
- Core Python is slow and not designed for numerical work.
- NumPy:
  - a powerful N-dimensional arrays, tools for integrating C/C++ and Fortran code, and useful linear algebra, Fourier transform, and random number capabilities.
- However, not always the fastest (an example will follow).

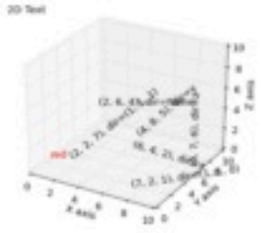
Visualising Data

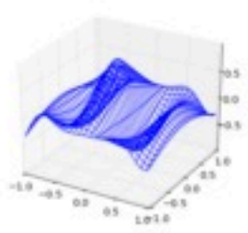
- Data visualisation is often crucial when trying to understand the data.
- Many Python packages like PyChar, igraph, graphviz, NetworkX, or you can call GnuPlot, Grace, etc. directly from Python.
- Will demonstrate one called matplotlib.

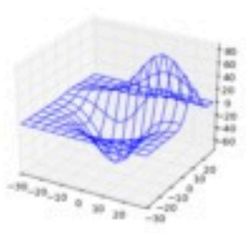
- Publication quality figures:
  - Supports e.g. LaTeX rendering.
  - Fonts, axes, etc. easily controllable.
  - Multiple output formats (ps, eps, PDF, ...)
- Resembles MATLAB in syntax.
- Can also be embedded to a GUI.
- Multiple backends: GTKAgg, QT4Agg, PDF...
- Active development community.

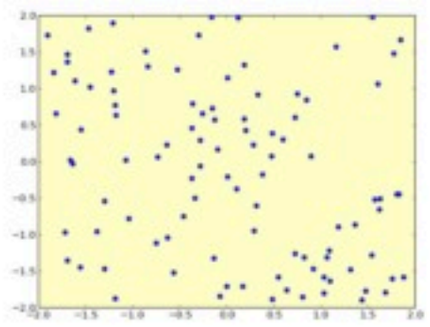


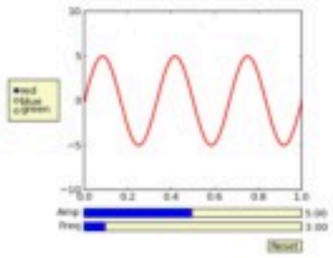


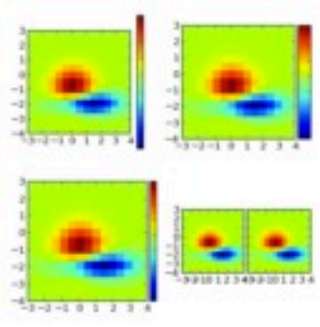

2D Text

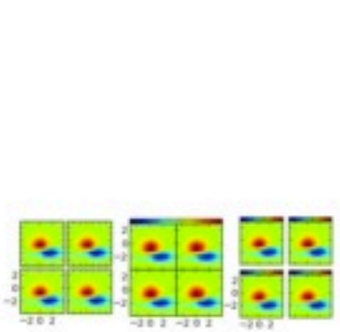







red
blue
green
Amp
Freq
5.00
3.00
Reset





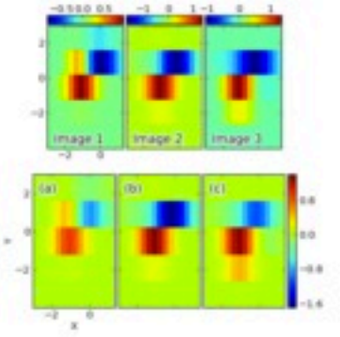
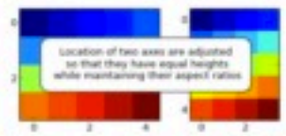
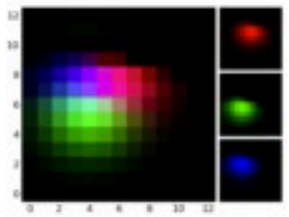
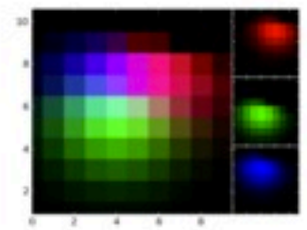
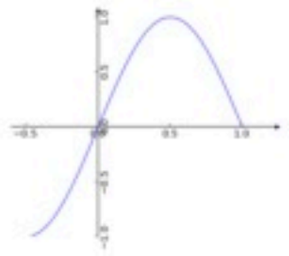


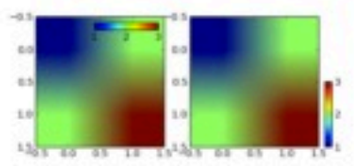
image 1
image 2
image 3
(a)
(b)
(c)

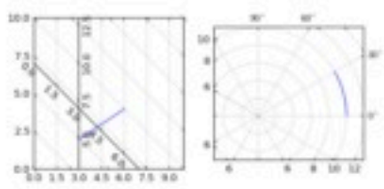

Location of two axes are adjusted
so that they have equal heights
while maintaining their aspect ratios

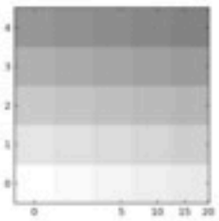


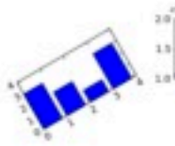


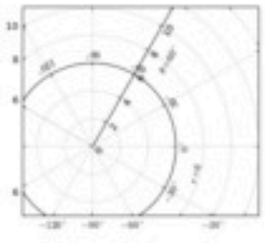


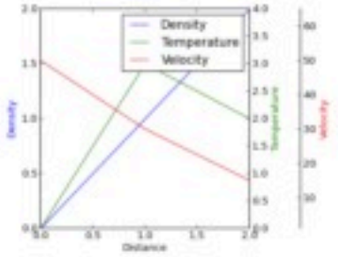


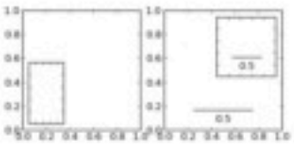


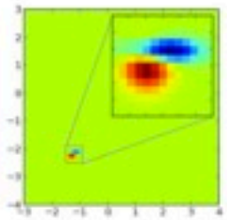


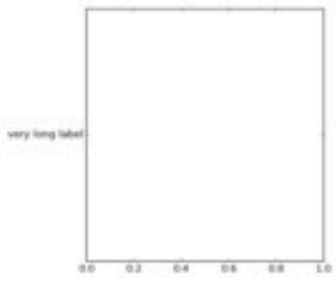


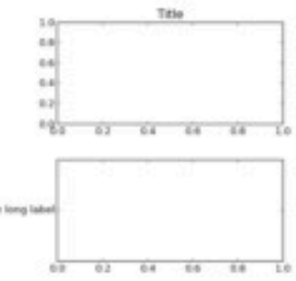


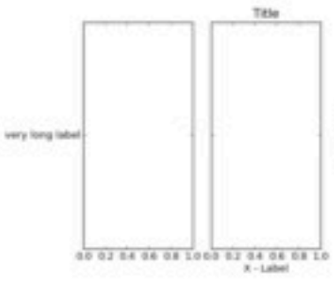

Density
Temperature
Velocity
Distance

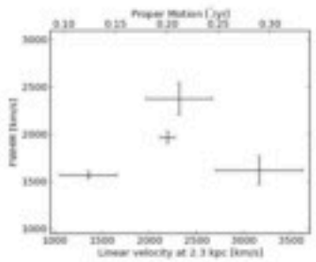


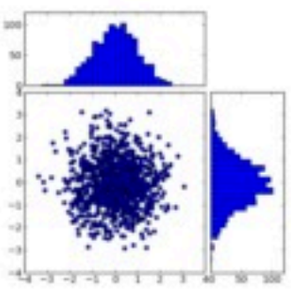


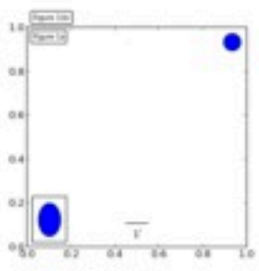

very long label

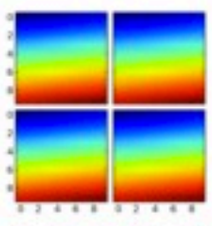

Title
very long label

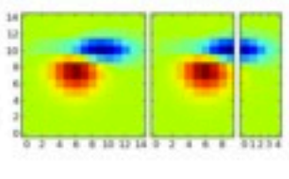

Title
very long label
X - Label

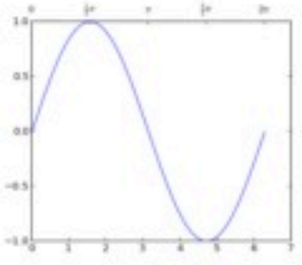

Proper Motion (''/yr)
Linear velocity at 2.3 kpc (km/s)

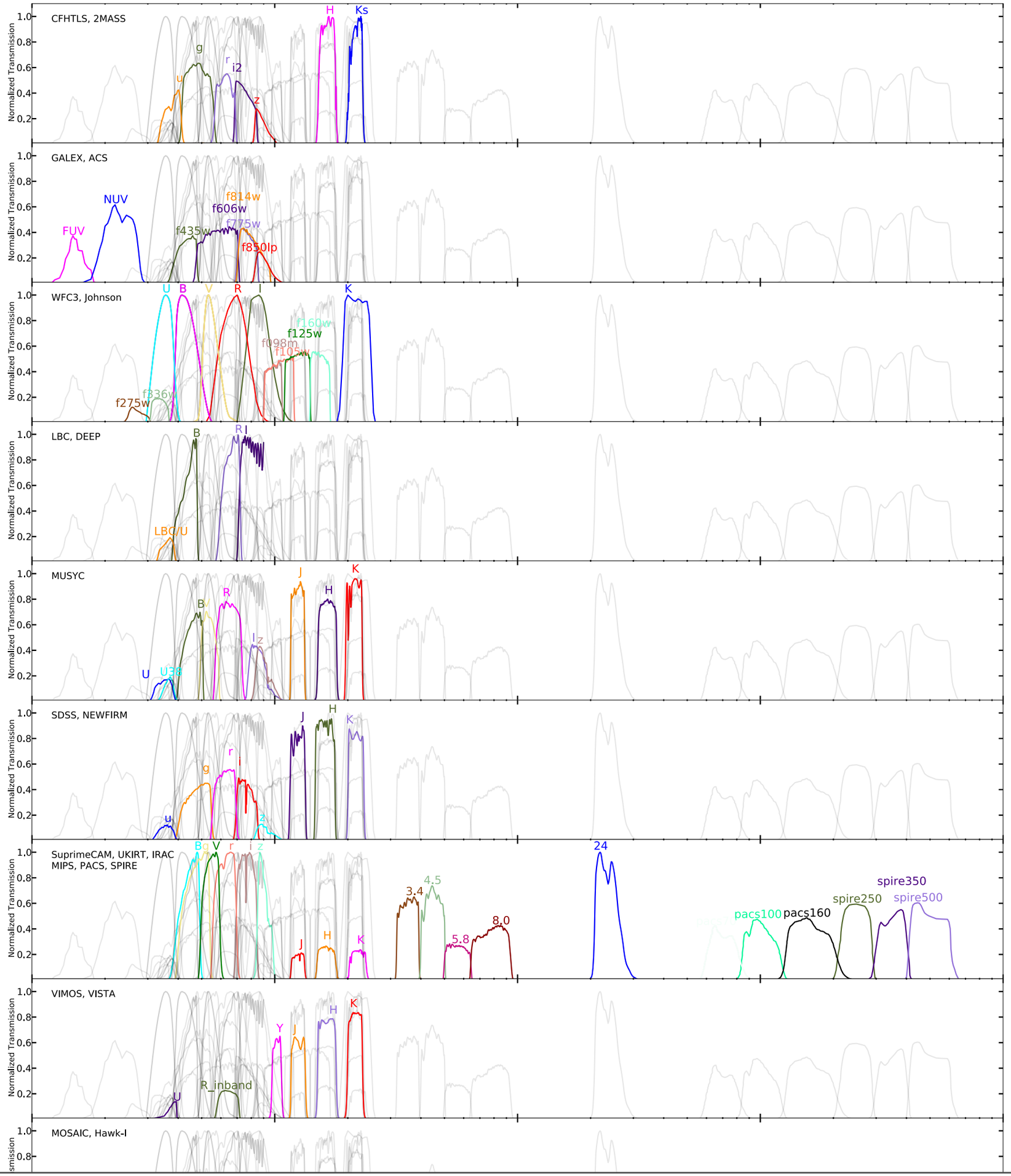


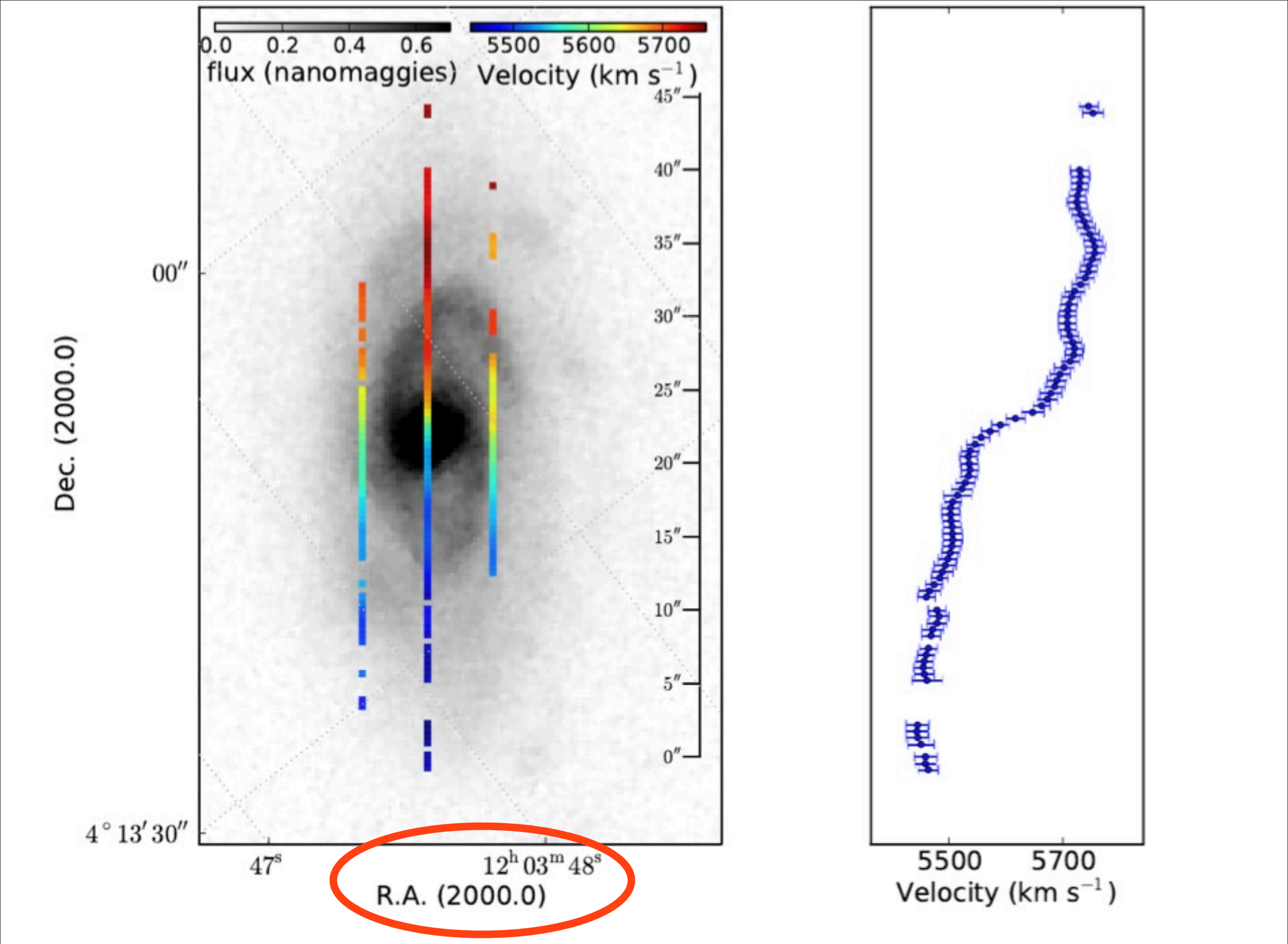




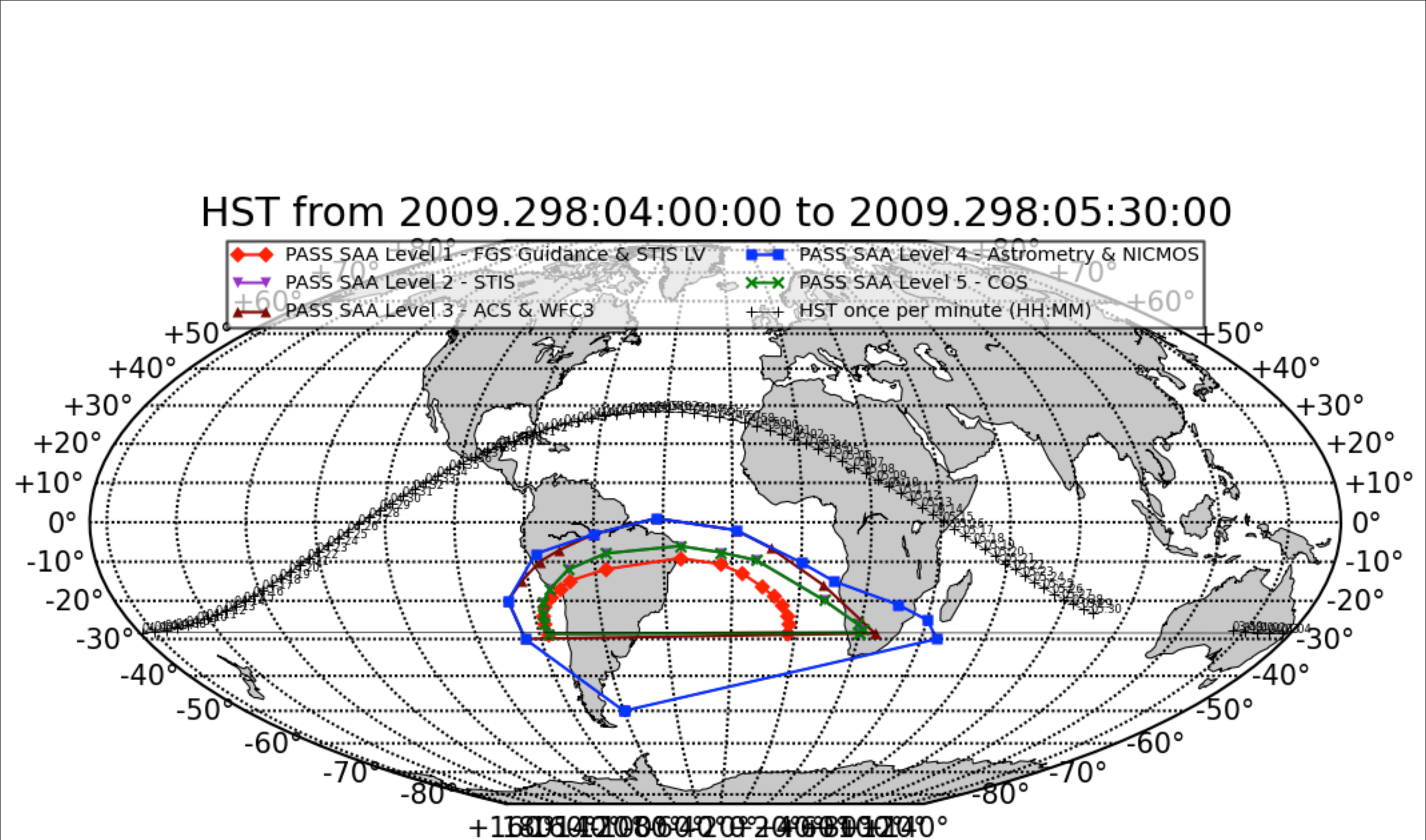


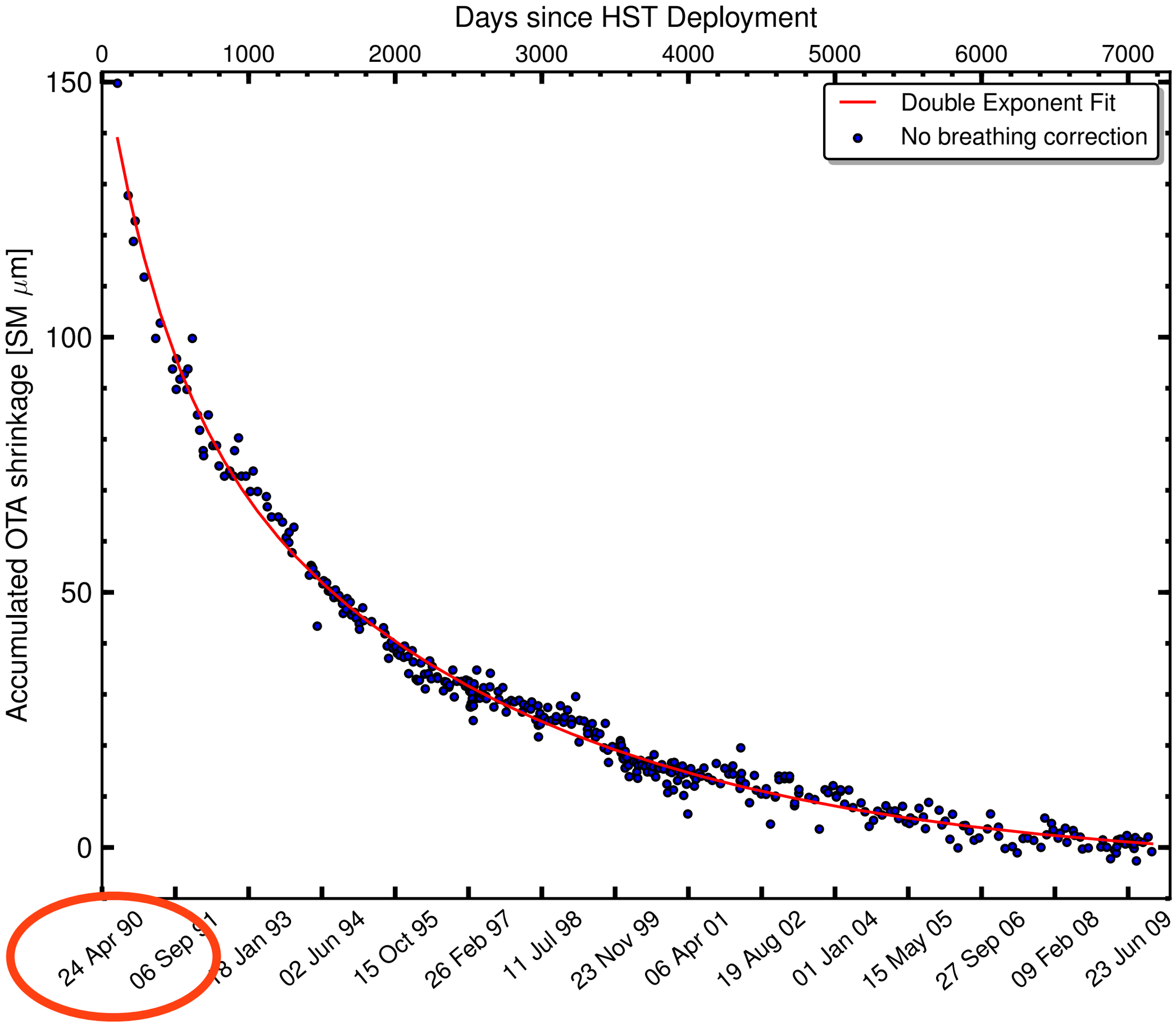



CFHTLS, 2MASS
u
g
r
i2
z
H
Ks
GALEX, ACS
FUV
NUV
f435w
f606w
f814w
f775w
f850lp
WFC3, Johnson
U
B
V
R
I
K
f275w
f336w
f098m
f105w
f125w
f160w
LBC, DEEP
B
R
I
LBC_U
MUSYC
U
U38
B
V
R
I
z
J
H
K
SDSS, NEWFIRM
u
g
r
i
z
J
H
K
SuprimeCAM, UKIRT, IRAC
MIPS, PACS, SPIRE
Bg
V
r
i
z
J
H
K
3.4
4.5
5.8
8.0
24
pacs100
pacs160
spire250
spire350
spire500
VIMOS, VISTA
U
R_inband
Y
J
H
K
MOSAIC, Hawk-I
Normalized Transmission
1.0
0.8
0.6
0.4
0.2

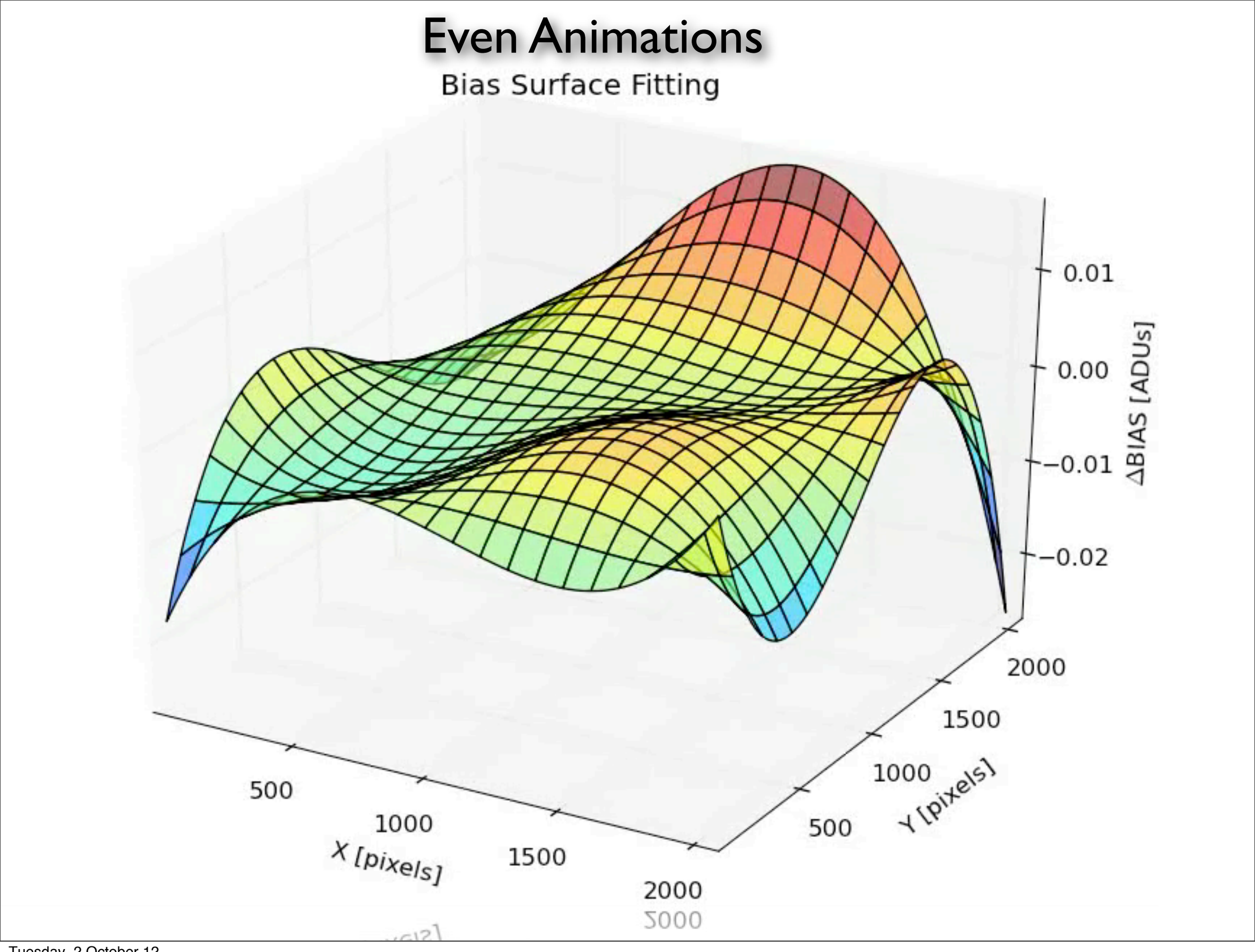

0.0 0.2 0.4 0.6
flux (nanomaggies)
5500 5600 5700
Velocity (km s^{-1})
45″
40″
35″
30″
25″
20″
15″
10″
5″
0″
00″
4° 13′ 30″
47^{s}
12^{h} 03^{m} 48^{s}
R.A. (2000.0)
Dec. (2000.0)
5500
5700
Velocity (km s^{-1})

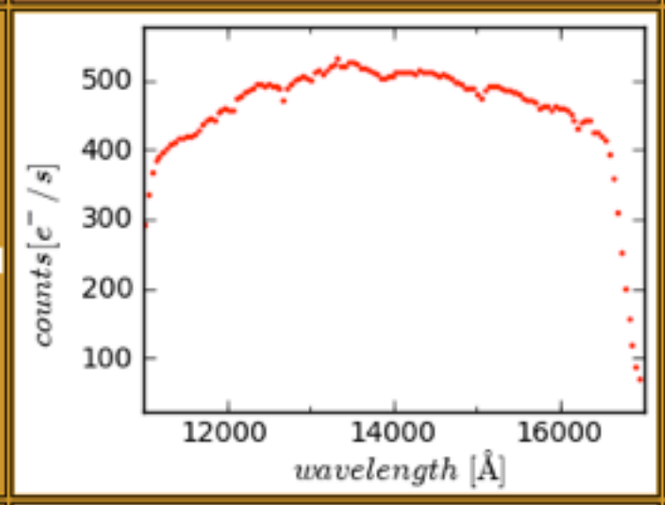
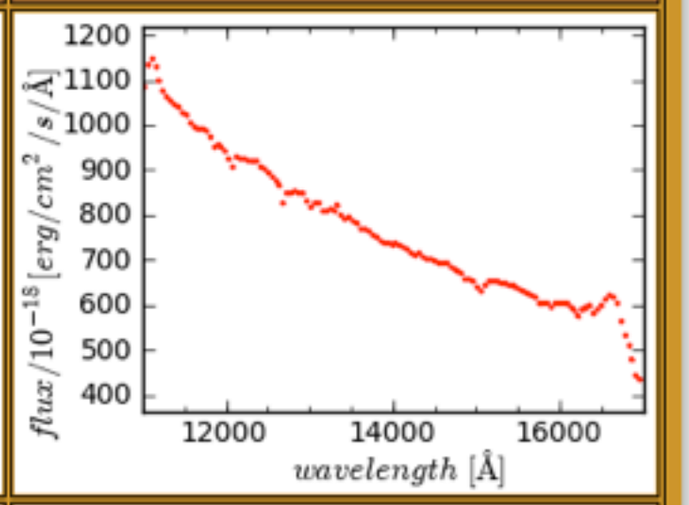
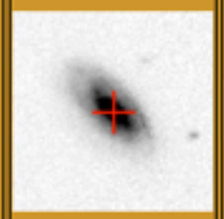
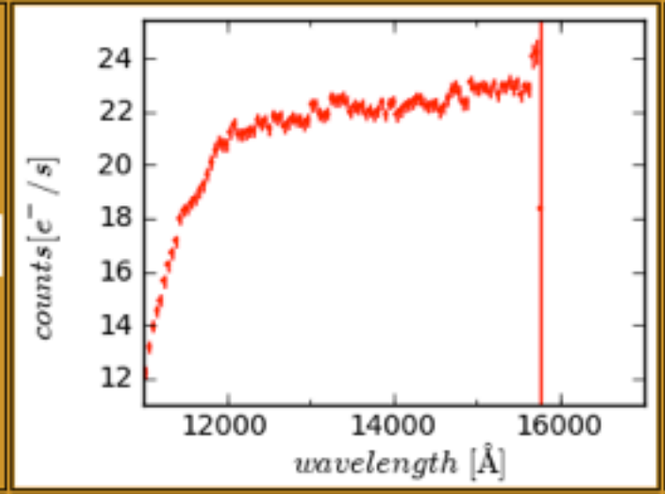
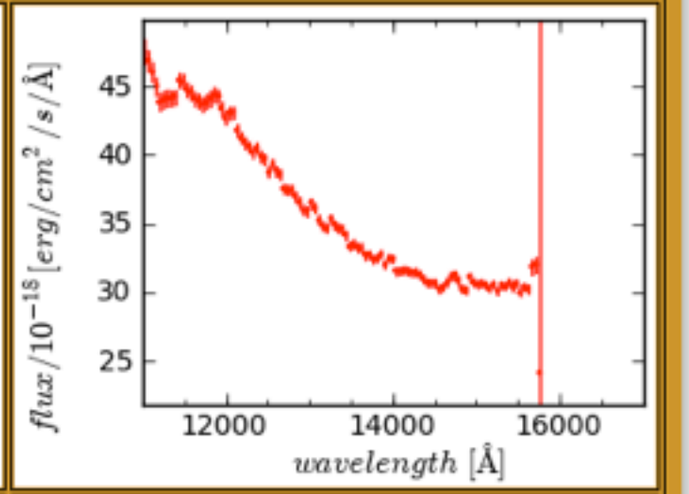
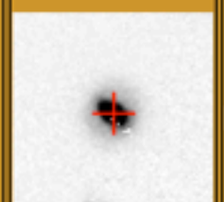
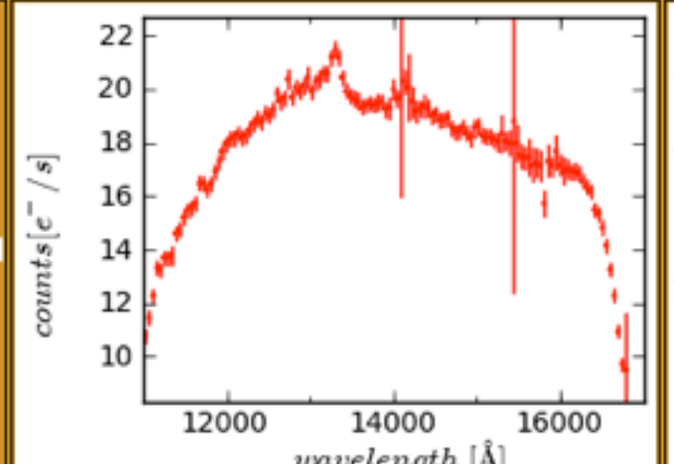
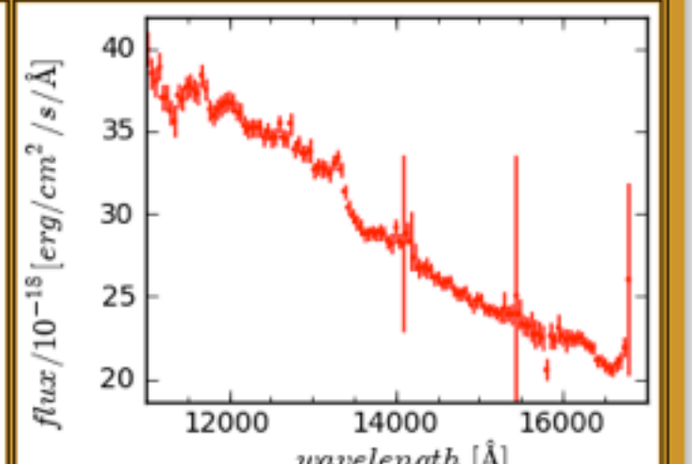
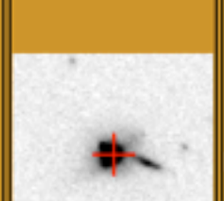
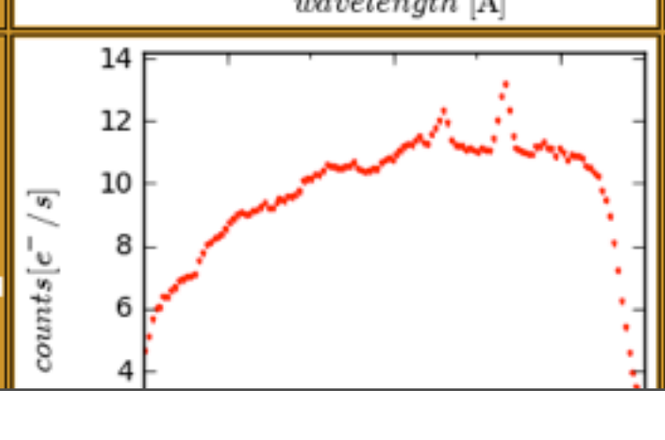
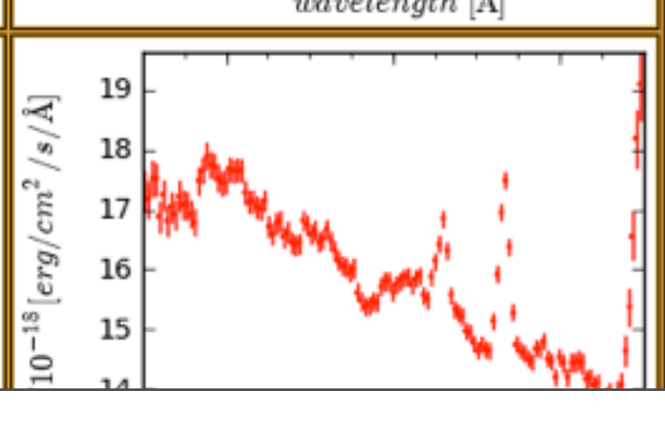
HST from 2009.298:04:00:00 to 2009.298:05:30:00



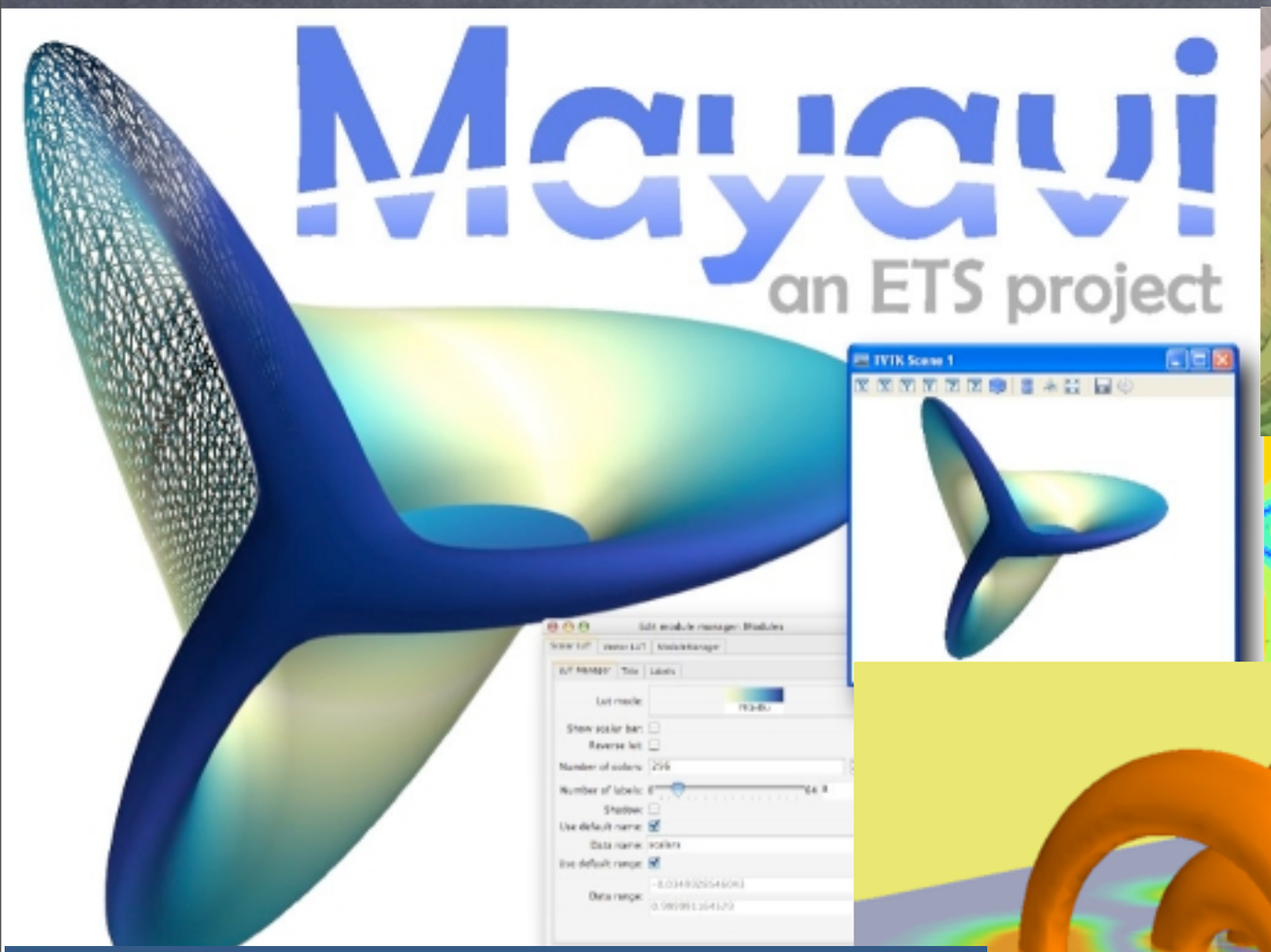

Days since HST Deployment
0
1000
2000
3000
4000
5000
6000
7000
150
100
50
0
Accumulated OTA shrinkage [SM μm]
Double Exponent Fit
No breathing correction
24 Apr 90
06 Sep 91
18 Jan 93
02 Jun 94
15 Oct 95
26 Feb 97
11 Jul 98
23 Nov 99
06 Apr 01
19 Aug 02
01 Jan 04
15 May 05
27 Sep 06
09 Feb 08
23 Jun 09

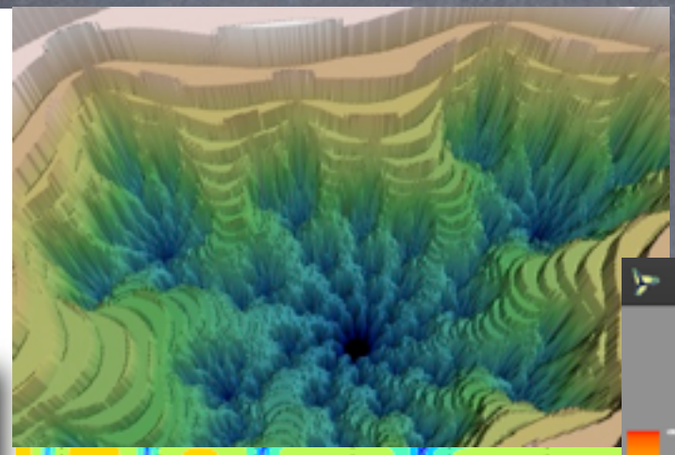
Even Animations

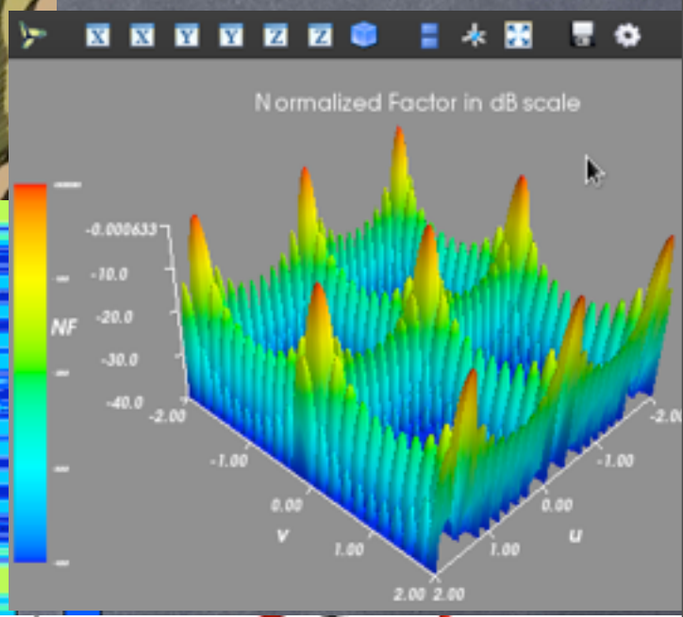


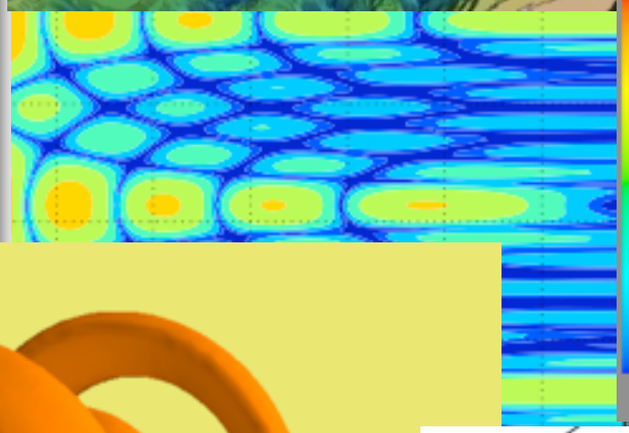
Seq. no.	Obj. no.	Magnitude mag(1392)	Image-coordinates	Direct image	Grism stamp	Raw spectrum	Calibrated spectrum
1	271	16.9052	x = 334.89 y = 273.611 ra: 3:32:13.78 dec: -27:42:14.2 POINT_SOURCE = 0.00				
2	315	18.1951	x = 1067.518 y = 145.746 ra: 3:32:20.24 dec: -27:41:32.7 POINT_SOURCE = 0.00				
3	1	18.239	x = 1041.393 y = 61.088 ra: 3:32:19.78 dec: -27:41:23.2 POINT_SOURCE = 0.00				
4	210	19.0157	x = 220.17 y = 462.168 ra: 3:32:13.21 dec: -27:42:41.5 POINT_SOURCE =				

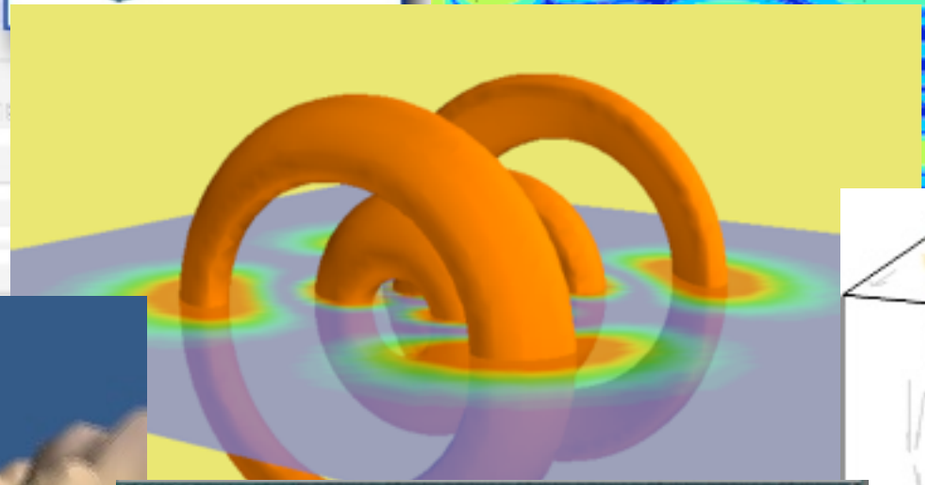
3D Visualization: mayavi

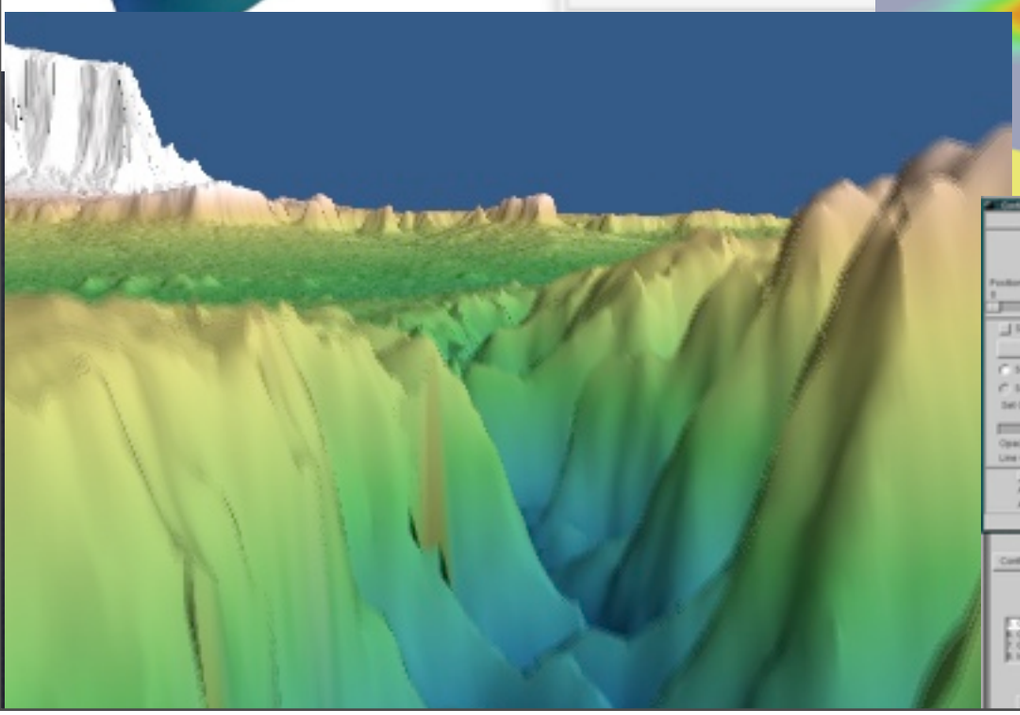


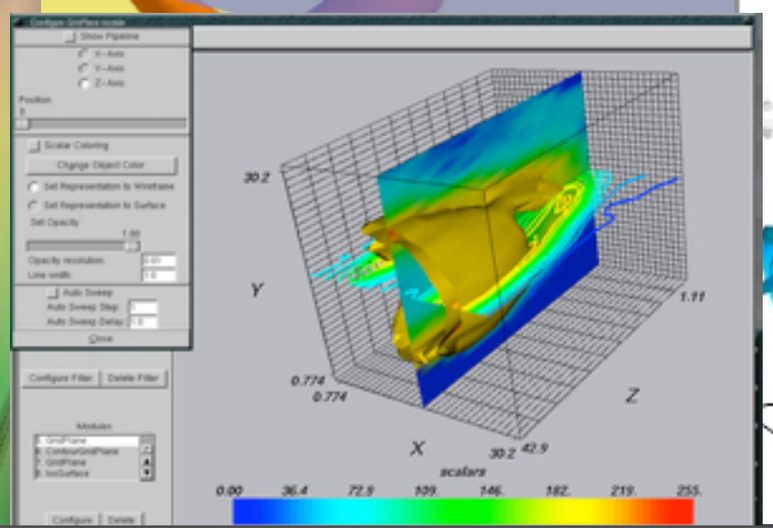


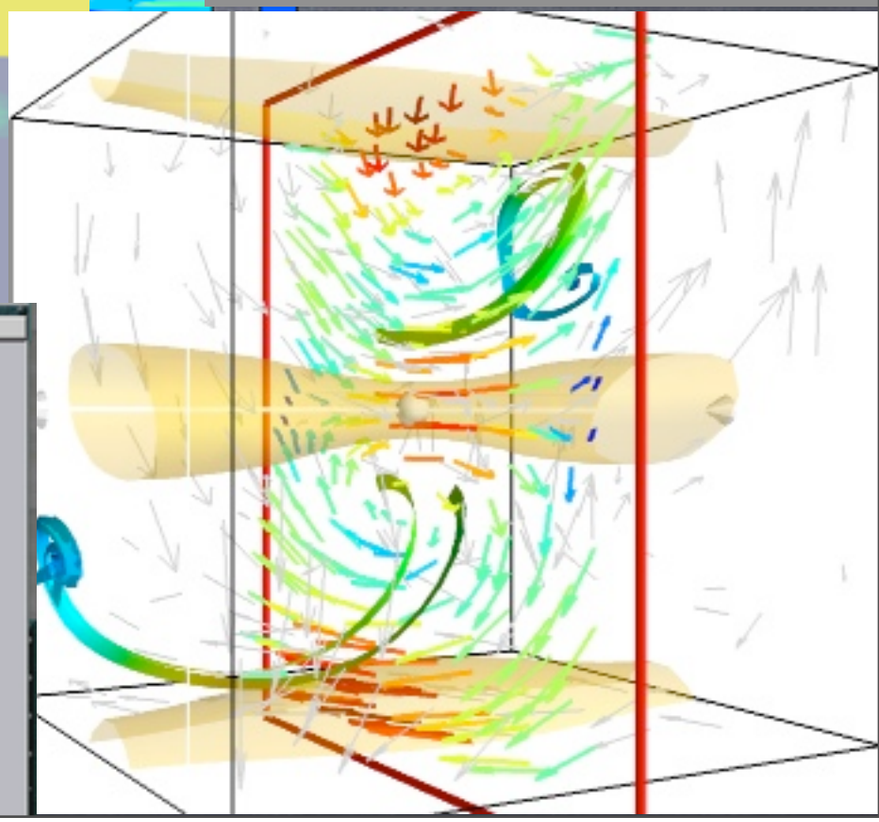












Go to Numerical Arrays Notebook

Astronomy Specific

- PyFITS
  - Provides an interface to FITS files
- PyWCS (I will import it from astropy)
  - Provides World Coordinate transformations following the SIP conventions.
- PyRAF
  - Gives users the ability to run IRAF tasks in an environment that has all the power and flexibility of Python.

Demonstrating IRAF calling

```
import pyraf
from pyraf import iraf
from iraf import stsdas, hst_calib, acs, calacs, images, immatch, sregister, blkavg
import acstools

try:
    #different naming convention in linux and macosx
    import drizzlepac as a
except:
    import astrodither as a
try:
    #different naming convention in linux and macosx
    from drizzlepac import astrodrizzle, tweakreg, pixtopix
except:
    from astrodither import astrodrizzle, tweakreg, pixtopix



class reduceACSWFCpoli():
    """
    This class provides methods for reducing ACS WFC polarimetry data.
    Uses astrodrizzle to combine images.
    """
```

```
def runCalACS(self):
    """
    Calls calACS and processes all given files or associations.
    """
    for f in self.input:
        calacs.run(input=f)

    #remove the raw files
    for f in glob.glob('*_raw.fits'):
        os.remove(f)
```

```
def initialProcessing(self):
    """
    Does the initial processing as follows:

        1. use astrodrizzle to combine images of the same POL filter using
           the shifts file generated by the tweakreg at point one.

    """
    #run astrodrizzle separately for each POL
    kwargs = dict(final_pixfrac=1.0, final_fillval=1.0, preserve=False,
                  updatewcs=True, final_wcs=False, build=True, skysub=False)
    for f in self.input:
        astrodrizzle.AstroDrizzle(input=f, mdriztab=False, editpars=False, **kwargs)
```

```
def doFinalDrizzle(self):
    """
    Does the final drizzling.

    :return: None
    """
    #we can now perform the final drizzle to drizzle all FLT images to POL frames
    kwargs = {'final_pixfrac': 1.0, 'skysub': False,
              'final_outnx': 2300, 'final_outny': 2300,
              'final_ra': 128.8369, 'final_dec': -45.1791,
              'updatewcs': False, 'final_wcs': True, 'preserve': False,
              'build': True, 'final_fillval': 1.0, #'final_wht_type': 'ERR',
              'final_refimage': 'jbj901akq_flt.fits[1]'}
    for f in self.input:
        astrodrizzle.AstroDrizzle(input=f, mdriztab=False, editpars=False, **kwargs)
```

Go to Notebook

Scientific Analysis

SciPy: numerical algorithms galore

- **linalg** : Linear algebra routines (including BLAS/LAPACK)
- **sparse** : Sparse Matrices (including UMFPACK, ARPACK,...)
- **fftpack** : Discrete Fourier Transform algorithms
- **cluster** : Vector Quantization / Kmeans
- **odr** : Orthogonal Distance Regression
- **special** : Special Functions (Airy, Bessel, etc).
- **stats** : Statistical Functions
- **optimize** : Optimization Tools
- **maxentropy** : Routines for fitting maximum entropy models
- **integrate** : Numerical Integration routines
- **ndimage** : n-dimensional image package
- **interpolate** : Interpolation Tools
- **signal** : Signal Processing Tools
- **io** : Data input and output

Scientific Analysis

SciPy: numerical algorithms galore

- **linalg** : Linear algebra routines (including BLAS/LAPACK)
- **sparse** : Sparse Matrices (including UMFPACK, ARPACK,...)
- **fftpack** : Discrete Fourier Transform algorithms
- **cluster** : Vector Quantization / Kmeans
- **odr** : Orthogonal Distance Regression
- **special** : Special Functions (Airy, Bessel, etc).
- **stats** : Statistical Functions
- **optimize** : Optimization Tools
- **maxentropy** : Routines for fitting maximum entropy models
- **integrate** : Numerical Integration routines
- **ndimage** : n-dimensional image package
- **interpolate** : Interpolation Tools
- **signal** : Signal Processing Tools
- **io** : Data input and output

scipy.io

- Loading and saving files, supports:
  - reading IDL savefiles
  - loading and saving MATLAB files
  - etc.
- Will demonstrate some methods later.

Go to Notebook

Data Access

- Reading and Writing Ascii files
  - Many possibilities; pure Python, SciPy, NumPy, asciitable, AstroAsciiData, BeautifulSoup...
- Databases
  - More convenient than Ascii files and not necessarily more complicated.
  - Will demonstrate:
    - Creating a database and querying it;
    - Accessing SDSS DR8 database.

Data Processing

- An example how to do a simple cosmic ray rejection and background subtraction to your STIS spectroscopic data.
  - Will use median filtering and polynomial fitting.

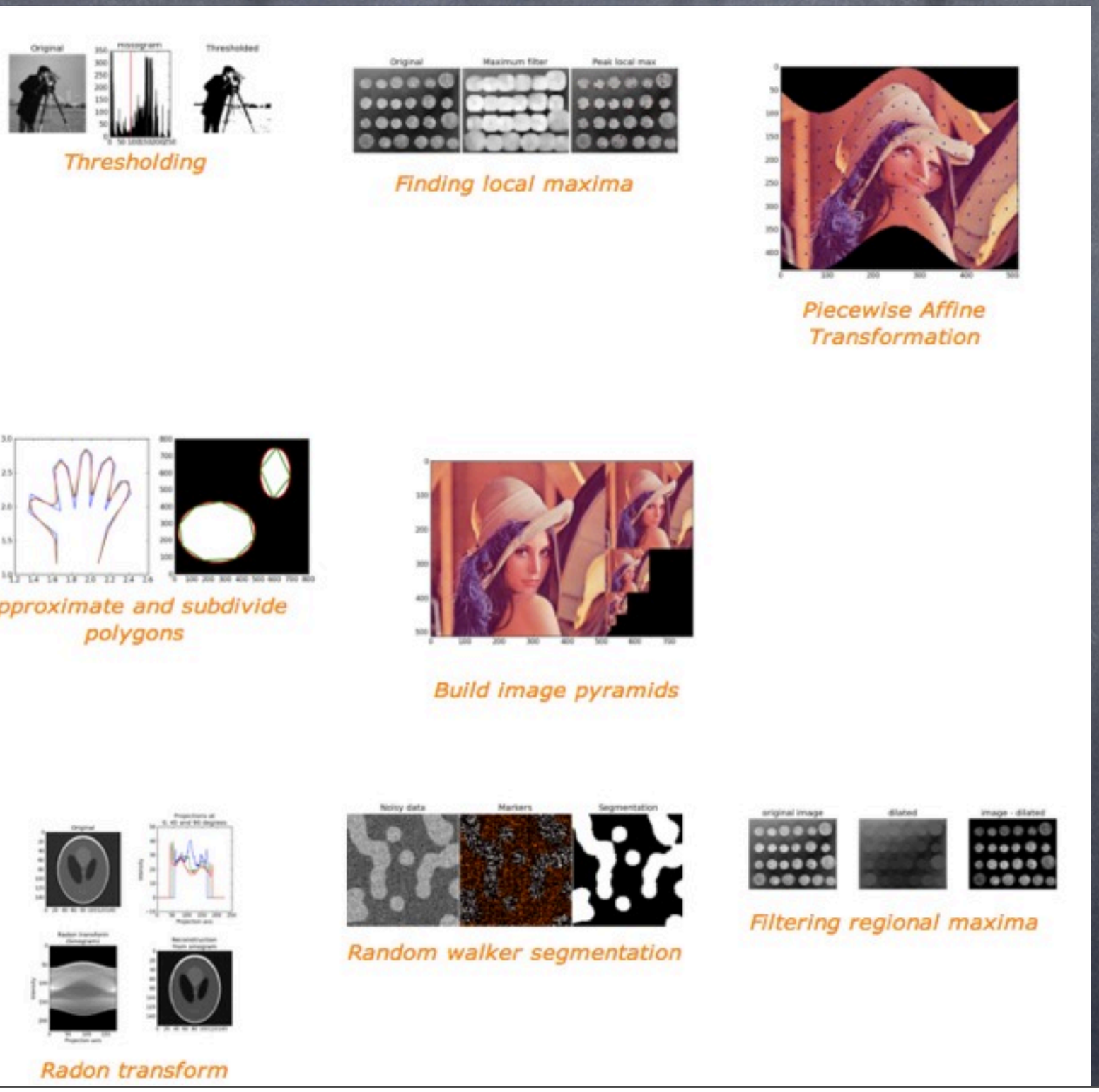
Go to Notebook

Image Processing

- Images can be considered as 2D arrays.
- scipy.ndimage:
  - filters: convolution, Gaussian, Laplace, Sobel, maximum, minimum, percentile...
  - interpolations: rotation, zoom, map_coorinates, arbitrary transform.
  - measurements: centre-of-mass, extrema, etc.
  - morphologies: dilation, erosion, opening, closing, etc.

Image Processing

- scikits-image: extends scipy.ndimage



Will show a few scikits examples, a detailed treatment of the topic would take weeks...

Demo Session

- Symbolic Calculus?
- Principal Component Analysis?
- MCMC fitting?
- Clustering?
- Outlier Detection?
- B-Spline Fitting?
- Source Finding and Photometry?

Speeding Things Up

- Already encountered:
  - NumPy instead of looping Python lists.
    - vectorizing functions (numpy.vectorize)
  - numexpr instead of NumPy
- Multiprocessor programming.
- Using GPUs instead of CPUs, for example, PyCUDA.
- Writing C or FORTRAN methods to support Python scripts.

Go to Notebook

Further Reading

- http://scipy-lectures.github.com/_downloads/PythonScientific-simple.pdf
- http://stsdas.stsci.edu/perry/pydatatut.pdf
- http://python4mpia.github.com/intro/quick-tour.html
- http://minrk.github.com/scipy-tutorial-2011/
- http://python4astronomers.github.com/index.html
- https://github.com/profjsb/python-bootcamp